IN THE CLAIMS:

This listing of claims will replace all prior versions, and listings, of claims in the application:

1. (Original): A method for data lookup for firewalling and network address translating, comprising:

instantiating a first data structure and a second data structure;

populating the first data structure with state information for a packet;

populating the second data structure with packet information for the packet; and

cross-linking the first data structure and the second data structure, the cross-linking including,

generating an index for the packet information; and

storing in the first data structure the index in association with the state information.

2. (Original): The method, according to claim 1, wherein the first data structure is a connection table.

3. (Original): The method, according to claim 1, wherein the second data structure is a network address translation table.

4. (Original): The method, according to claim 1, wherein the state information includes Transport Control Protocol state.

5. (Original): The method, according to claim 1, wherein the packet information comprises a five-tuple of information.

6. (Original): The method, according to claim 1, wherein the generating comprises hashing at least a portion of the packet information to generate the index for the packet information.

7. (Original): A method for creating data structures for physical layer addressing, comprising:

instantiating a first, a second and a third data structure;

populating the first data structure with state information;

populating the second data structure with network address translation information;

populating the third data structure with interface information; and

cross-linking the first data structure and the second data structure to the third data structure, the cross-linking including:

generating an index for the interface information; and

storing the index in the first data structure in association with the state information and in the second data structure in association with the network address translation information.

8. (Original): The method, according to claim 7, further comprising:

instantiating a fourth data structure;

populating the fourth data structure with routing information; and

cross-linking the fourth data structure to the third data structure, the cross-linking including:

storing the index in the fourth data structure in association with the routing information.

9. (Original): The method, according to claim 8, wherein the third data structure is an address resolution table.

10. (Original): The method, according to claim 9, wherein the first data structure is a connection table.

11. (Original): The method, according to claim 10, wherein the populating of the first data structure comprises adding connection information to the first data structure.

12. (Original): The method, according to claim 10, wherein the connection information comprises a packet five-tuple of information.

13. (Original): The method, according to claim 10, wherein the second data structure is a network address translation table.

14. (Original): The method, according to claim 13, wherein the fourth data structure is a routing table.

15. (Original): The method, according to claim 14, wherein the generating comprises hashing at least a portion of the interface information.

16. (Original): The method, according to claim 15, wherein the interface information comprises a media access control address.

17. (Original): The method, according to claim 16, wherein the interface information comprises a virtual local area network identifier.

18. (Original): The method, according to claim 17, wherein the interface information comprises an interface mask.

19. (Original): A method for security protocol support, comprising:

creating a table, the table including a first, a second and a third assigned data space;

populating the first assigned data space to indicate that a security protocol is being used;

populating the second assigned data space with a first portion of a security protocol string; and

populating the third assigned data space with a second portion of the security protocol string.

20. (Original): The method, according to claim 19, wherein the table is a connection table.

21. (Original): The method, according to claim 20, wherein the connection table comprises a fourth data space for transport control protocol state information, the method further comprising populating the fourth data space with a transport control protocol state.

22. (Original): The method, according to claim 20, wherein the second assigned data space is a remote port data space, and wherein the third assigned data space is a local port data space.

23. (Original): The method, according to claim 19, wherein the table is a network address translation table.

24. (Original): The method, according to claim 23, wherein the second assigned data space is a remote port data space, and wherein the third assigned data space is a public port data space.

25. (Original): A method for creating at least one data structure, comprising:

determining if a firewall is activated;

determining if network address translator is activated; and

creating the at least one data structure responsive to one of:

the firewall and the network address translator being activated;

the firewall being activated and the network address translator not being activated; and

the firewall not being activated and the network address translator being activated.

26. (Original): A method for creating data structures, comprising:

determining if a firewall is activated;

determining if network address translator is activated; and

creating a connection table responsive to whether the firewall is activated; and

creating the connection table and a network address translation table responsive to whether the network address translator is activated.

27. (Original): The method, according to claim 26, further comprising:

creating an address resolution table; and

creating a routing table.

28. (Original): A data structure for routing packets, comprising:

an Internet Protocol destination address data space for storing Internet Protocol destination addresses;

an Internet Protocol source address data space for storing Internet Protocol source address; and

an address resolution table index data space for storing indices to an address resolution table;

wherein the address resolution table includes a media access control address data space for storing media access control addresses.

29. (Original): The data structure, according to claim 28, wherein the data structure may be used exclusively in combination with the address resolution table for the routing of packets when both firewalling and network address translating are not present.

30. (Original): A data structure for routing packets, consisting of:

an Internet Protocol destination address data space for storing Internet Protocol destination addresses;

an Internet Protocol source address data space for storing Internet Protocol source address; and

an address resolution table index data space for storing indices to an address resolution table.

31. (Original): A method of forming hashing table chains, comprising:

obtaining a first connection hash value, the first connection hash value pointing to a first slot in the hashing table;

obtaining a second connection hash value, the second connection hash value pointing to the first slot in the hashing table;

assigning the second connection hash value to a second slot in the hashing table;

pointing the first slot toward the second slot;

obtaining a third connection hash value, the third connection hash value pointing to the second slot in the hashing table;

moving contents of the second slot to a third slot in the hashing table; and

assigning the third connection hash value to the second slot in the hashing table.

32. (Original): The method, according to claim 31, wherein the first connection hash value, the second connection hash value and the third connection hash value are respective hashes of packet information.

33. (Original): The method, according to claim 32, wherein the packet information is a packet five-tuple.

34. (Original): A method for tracking packet states, comprising:

initiating tracking of state from a CLOSED state;

from the CLOSED state, tracking transition to a LISTEN state or a SYN-SENT state;

from the LISTEN state, tracking transition to one of the CLOSED state, a SYN-RCVD state or the SYN-SENT state;

from the SYN-RCVD state, tracking transition to either a first hardware state or a SYN-RCVD-SYN-SENT state;

from the SYN-SENT state, tracking transition to either a second hardware state or the SYN-RCVD-SYN-SENT state;

from the SYN-RCVD-SYN-SENT state, tracking transition to either a first SYN-RCVD-SYN-SENT-ACK state or a second SYN-RCVD-SYN-SENT-ACK state; and

from either the first SYN-RCVD-SYN-SENT-ACK state or the second SYN-RCVD-SYN-SENT-ACK state, tracking transition to a third hardware state.

35. (Original): The method, according to claim 34, wherein the transition from the LISTEN state to either the SYN-RCVD state or the SYN-SENT state is respectively responsive to detecting either a received SYN or a sent SYN for a packet.

36. (Original): The method, according to claim 34, wherein the transition from the SYN-RCVD state to either the first hardware state or the SYN-RCVD-SYN-SENT state is respectively responsive to detecting either a sent SYN-ACK or a sent SYN for a packet.

37. (Original): The method, according to claim 34, wherein the transition from the SYN-SENT state to either the second hardware state or the SYN-RCVD-SYN-SENT state is respectively responsive to detecting either a received SYN-ACK or a received SYN for a packet.

38. (Original): The method, according to claim 34, wherein the transition from the SYN-RCVD-SYN-SENT state to either the first SYN-RCVD-SYN-SENT-ACK state or the second SYN-RCVD-SYN-SENT-ACK state is respectively responsive to detecting either a sent SYN-ACK or a received SYN-ACK for a packet.

39. (Original): The method, according to claim 34, wherein the transition from either the first SYN-RCVD-SYN-SENT-ACK state or the second SYN-RCVD-SYN-SENT-ACK state to the third hardware state is respectively responsive to detecting either a received SYN-ACK or a sent SYN-ACK for a packet.

40. (Original): The method, according to claim 34, wherein the first hardware state is a SYN-RCVD-SYN-ACK-SENT state, the second hardware state is SYN-SENT-SYN-ACK-RCVD state, and the third hardware state is a connection-established state.

41. (Original): The method, according to claim 34, wherein the transition from the LISTEN state to the CLOSED state is responsive to an age out condition for a packet.

42. (Original): The method, according to claim 34, wherein the transition from the LISTEN state to the CLOSED state is responsive to a close condition for a packet.

43. (Original): The method, according to claim 34, wherein the transition from the CLOSED state to the SYN-SENT state is responsive to a sent SYN for a packet.

44. (Original): The method, according to claim 34, wherein the LISTEN state, the SYN-RCVD state, the SYN-SENT state, the SYN-RCVD-SYN-SENT state, the first SYN-RCVD-SYN-SENT-ACK state and the second SYN-RCVD-SYN-SENT-ACK state are software states.

45. (Original): An apparatus for tracking packet states, comprising:

means for initiating tracking of state from a first CLOSED state;

means for tracking software states for packets to one of a first, a second and a third hardware state, the first hardware state being a SYN-RCVD-SYN-ACK-SENT state, the second hardware state being SYN-SENT-SYN-ACK-RCVD state, and the third hardware state being a connection-established state; and

means for tracking hardware states for the packets including:

means for tracking transition to the connection-established state from the SYN-RCVD-SYN-ACK-SENT state;

means for tracking transition to the connection-established state from the SYN-SENT-SYN-ACK-RCVD state;

means for tracking transition to a first FIN-WAIT state from the SYN-RCVD-SYN-ACK-SENT state, the SYN-SENT-SYN-ACK-RCVD state or the connection-established state; and

means for tracking transition to a CLOSE-WAIT-FIN state from the SYN-RCVD-SYN-ACK-SENT state, the SYN-SENT-SYN-ACK-RCVD state or the connection-established state.

46. (Original): The apparatus, according to claim 45, further comprising:

means for tracking transition to a second FIN-WAIT state, a FIN-WAIT-FIN state or a CLOSING-FIN state from the first FIN-WAIT state; and

means for tracking transition to the CLOSING-FIN state, a LAST-ACK state or a CLOSE-WAIT state from the CLOSE-WAIT-FIN state.

47. (Original): The apparatus, according to claim 46, further comprising:

means for tracking transition to the FIN-WAIT-FIN state from the second FIN-WAIT state;

means for tracking transition to the FIN-WAIT-FIN state or a CLOSING state from the CLOSING-FIN state; and

means for tracking transition to the LAST-ACK state from the CLOSE-WAIT state.

48. (Original): The apparatus, according to claim 47, further comprising:

means for tracking transition to a TIME-WAIT state from the FIN-WAIT-FIN state;

means for tracking transition to the TIME-WAIT state from the CLOSING state; and

means for tracking transition to a second CLOSED state from the LAST-ACK state or the TIME-WAIT state.

49. (Original): The apparatus, according to claim 45, wherein the transition to the first FIN-WAIT state from the SYN-RCVD-SYN-ACK-SENT state, the SYN-SENT-SYN-ACK-RCVD state or the connection-established state is responsive to a sent FIN.

50. (Original): The apparatus, according to claim 45, wherein the transition to the connection-established state from the SYN-RCVD-SYN-ACK-SENT state is responsive to a received ACK of a SYN.

51. (Original): The apparatus, according to claim 45, wherein the transition to the connection-established state from the SYN-SENT-SYN-ACK-RCVD state is responsive to a sent ACK of a SYN.

52. (Original): The apparatus, according to claim 46, wherein the transition to the second FIN-WAIT state from the first FIN-WAIT state is responsive to a received ACK of a FIN.

53. (Original): The apparatus, according to claim 46, wherein the transition to the FIN-WAIT-FIN state from the first FIN-WAIT state is responsive to a received FIN and a received ACK of the received FIN in a packet.

54. (Original): The apparatus, according to claim 46, wherein the transition to the CLOSING-FIN state from the first FIN-WAIT state is responsive to a received FIN.

55. (Original): The apparatus, according to claim 46, wherein the transition to the CLOSING-FIN state from the CLOSE-WAIT-FIN state is responsive to a sent FIN.

56. (Original): The apparatus, according to claim 46, wherein the transition to the LAST-ACK state from the CLOSING-WAIT-FIN state is responsive to a sent FIN and a sent ACK of the sent FIN in a packet.

57. (Original): The apparatus, according to claim 46, wherein the transition to the CLOSE-WAIT state from the CLOSE-WAIT-FIN state is responsive to a sent ACK of a FIN.

58. (Original): The apparatus, according to claim 47, wherein the transition to the FIN-WAIT-FIN state from the second FIN-WAIT state is responsive to a received FIN.

59. (Original): The apparatus, according to claim 47, wherein the transition to the FIN-WAIT-FIN state from the CLOSING-FIN state is responsive to a received ACK of a FIN.

60. (Original): The apparatus, according to claim 47, wherein the transition to the CLOSING state from the CLOSING-FIN state is responsive to a sent ACK of a FIN.

61. (Original): The apparatus, according to claim 47, wherein the transition to the LAST-ACK state from the CLOSE-WAIT state is responsive to a sent FIN.

62. (Original): The apparatus, according to claim 48, wherein the transition to a TIME-WAIT state from the FIN-WAIT-FIN state is responsive to a sent ACK of a FIN.

63. (Original): The apparatus, according to claim 48, wherein the transition to the TIME-WAIT state from the CLOSING state is responsive to a received ACK of a FIN.

64. (Original): The apparatus, according to claim 48, wherein the transition to a second CLOSED state from the LAST-ACK state is responsive to a received ACK of a FIN.

65. (Original): The apparatus, according to claim 48, wherein the transition to a second CLOSED state from the TIME-WAIT state is responsive to a timed out condition.

66. (Original): A method for network protocol processing, comprising:

obtaining a packet for network address translation, the packet having a media access control header;

obtaining information, including the media access control header, from the packet;

parsing out the information into one or more data structures;

determining if a network processing unit is in a pass-through mode responsive to the media access control header; and

responsive to the network processing unit not being in the pass-through mode,

determining whether multicast or broadcast is active; and

determining whether a protocol type for the packet is supported by the network processing unit.

67. (Original): The method, according to claim 66, wherein the pass-through mode is a frame conversion only mode.

68. (Original): The method, according to claim 66, further comprising:

responsive to the network processing unit being in the pass-through mode,

converting header format of the packet to provide a composed packet; and

transmitting the composed packet.

69. (Original): The method, according to claim 68, wherein the composed packet is transmitted to a firewall module of the network processing unit.

70. (Original): The method, according to claim 69, wherein the composed packet is transmitted to an address translator portion of the network processing unit.

71. (Original): The method, according to claim 69, wherein the composed packet is transmitted to a sequence processor portion of the network processing unit.

72. (Original): The method, according to claim 66, further comprising determining whether a data link layer is valid.

73. (Original): The method, according to claim 66, wherein the determining whether multicast or broadcast is active further comprises determining whether a frame is for broadcast or multicast.

74. (Original): The method, according to claim 66, wherein the determining whether the protocol type for the packet is supported by the network processing unit comprises determining whether the protocol type of the packet is supported on an incoming interface.

75. (Original): An apparatus for network protocol processing, comprising:

means for obtaining a packet for network address translation, the packet having a media access control header;

means for obtaining information, including the media access control header, from the packet;

means for parsing out the information into one or more data structures;

means for determining if a network processing unit is not in a pass-through mode responsive to the media access control header;

means for determining whether multicast reception is active;

means for determining whether a data link layer is valid; and

means for determining whether a protocol type for the packet is supported by the network processing unit.

76. (Original): A signal-bearing medium containing a program which, when executed by a processor, causes execution of a method for network protocol processing comprising:

obtaining a packet for network address translation, the packet having a media access control header;

obtaining information, including the media access control header, from the packet;

parsing out the information into one or more data structures;

determining if a network processing unit is not in a pass-through mode responsive to the media access control header;

determining whether multicast reception is active;

determining whether a data link layer is valid; and

determining whether a protocol type for the packet is supported by the network processing unit.